Amendments to the Claims:

This listing of claims will replace all prior versions, and listings, of claims in the application:

1. (Currently Amended) A method of allocating resources in a plurality of processors system, each processor of said plurality having a cache associated therewith, comprising:

when at least one processor of said plurality of processors system is idle, detecting at least one other processor of said plurality of processors which is not idle;

timing a predetermined period of time during which said at least one other processor which is not idle remains not idle; and

if said at least one other processor which is not idle remains not idle when the predetermined period of time has elapsed, then poaching a process on a queue of said at least one non-idle processor to be run by said at least one processor which is idle~~.~~, wherein if more than one processor is idle, poaching the process with the idle processor which is electrically closest to the at least one non-idle processor, wherein the time period during which a non-idle processor is allowed to remain non-idle is greater the farther away a non-idle processor is electrically located relative to an idle processor, and wherein if the processor electrically closest to the idle processor is idle, and if the processor electrically closest to it is idle, it will then try to poach from the next processor which is electrically closest to it until it encounters a non-idle processor on which poaching can occur.

2. (Canceled)

3. (Canceled)

4. (Canceled)

5. (Original) The method of Claim 1 wherein it is conducted on a ccNUMA system.

6. (Original) The method of Claim 1 wherein each processor starts a timer associated therewith when it goes non-idle, and allowing an idle processor to poach a process therefrom only when a predetermined amount of time has elapsed on the timer.

7. (Canceled)

8. (Canceled)

9. (Original) The method of Claim 1 wherein said processor system is a four block system comprised of sixteen processors arranged as CPU_0, CPU_1, CPU_2, CPU_3 on a locale 0, CPU_4, CPU_5, CPU_6, CPU_7 on a locale 1, CPU_8, CPU_9, CPU_{10}, CPU_{11} on a locale 2, and CPU_{12}, CPU_{13}, CPU_{14}, CPU_{15} on a locale 3, and further comprising having each idle processor attempt to poach first from its nearest non-idle processor in its locale, and if there are no non-idle processors in said locale, from a processor in the closest locale having non-idle processors.

10. (Original) The method of Claim 9 wherein in the event more than one idle processors attempt to poach a process from a non-idle processor, allowing the idle processor in closest electrical proximity to the non-idle processor poach the process.

11. (Currently Amended) A data processing system for allocating resources for simultaneously executing a plurality of processing tasks, comprising:

a plurality of processors, each having a cache associated therewith;

a timer associated with each processor for timing from the beginning of receiving a process from the processor's queue, the duration of time the process is run, during which time the processor is running the process is non-idle;

means for detecting at least one other processor of said plurality of processors which is not idle, and for determining the duration of time any one of said at least one other processor is not idle;

means for then poaching a process from a non-idle processor by an idle processor when said duration of time during which the non-idle processor is running a process exceeds a predetermined amount~~.~~; and

means for allowing an idle processor connected electrically closest connected to a non-idle processor to poach a process in the event there are more than one non-idle processors, further configured to allow the time period during which a processor is allowed to remain non-idle to be determined in accordance with proximity in electrical connection between the idle

processor and a non-idle processor, wherein the greater the connection distance, the greater the predetermined amount of time, and further configured for having an idle processor first try to poach from a non-idle processor electrically closest to it, and if the processor electrically closest to it is idle, then to try to poach from the next electrically closest processor until it encounters a non-idle processor on which poaching can occur.

12. (Canceled)

13. (Canceled)

14. (Canceled)

15. (Original) The system of Claim 11 wherein said system is a ccNUMA system.

16. (Original) The system of Claim 11 wherein each processor is configured for starting the timer associated therein when it goes non-idle.

17. (Original) The system of Claim 16 wherein each processor is configured for setting the predetermined amount of time in relation to electrical connection proximity between an idle processor and a non-idle processor.

18. (Original) The system of Claim 17 wherein each processor is configured for setting the predetermined amount of time such that the greater the electrical connection distance between an idle and a non-idle processor, the greater the amount of time which is allowed to elapse before an idle processor is allowed to poach a process from a non-idle processor.

19. (Original) The system of Claim 1 wherein the plurality of processors are arranged in four blocks comprised of sixteen processors arranged as CPU_0, CPU_1, CPU_2 and CPU_3 on a locale 0, CPU_4, CPU_5, CPU_6 and CPU_7 on a locale 1, CPU_8, CPU_9, CPU_{10} and CPU_{11} on a locale 2, and CPU_{12}, CPU_{13}, CPU_{14} and CPU_{15} on a locale 3, and comprising said processors configured such that each idle processor attempts to poach first from its nearest non-idle processor in its locale, and if there are no non-idle processors in said locale, from a processor in the closest locale having non-idle processors.

20. (Original) The system of Claim 19 wherein said processors are configured for allowing an idle processor in closest electrical connection to a non-idle processor to poach a process therefrom in the event more than one idle processor attempts to poach a process from said non-idle processor.

21. (Previously Presented) A method of allocating resources in a plurality of processors system, each processor of said plurality having a cache associated therewith, comprising:

when at least one processor of said plurality of processors of said system is idle, detecting at least one other processor of said plurality of processors which is not idle;

starting a timer at said at least one other processor when it goes not idle, and timing a predetermined period of time during which said at least one other processor which is not idle remains not idle;

if said at least one other processor which is not idle remains not idle when the predetermined period of time has elapsed on the timer, then poaching a process therefrom with an idle processor; and

conducting said poaching in a manner wherein an idle processor first tries to poach from a non-idle processor electrically closest to it, and if the processor electrically closest to the idle processor is idle, trying to poach from the next processor which is electrically closest to it until it encounters a non-idle processor on which poaching can occur, and wherein the predetermined time period during which a non-idle processor is allowed to remain non-idle is greater the farther electronically away a non-idle processor is located relative to an idle processor.